Search

- Lakeview Research home

ResourcePages
- Embedded Ethernet and Internet

- USB
  - USB Code
  - HID Page
- Serial Ports
- Parallel Port
- 8052-Basic Microcontrollers
- PC Boards

Books
- Embedded Ethernet and Internet Complete
- USB Complete
- Serial Port Complete
- Parallel Port Complete
- The Microcontroller Idea Book
- Making Printed Circuit Boards

Lakeview Research home > USB Central > Boarding the Universal Serial Bus

Boarding the Universal Serial Bus: Pick the Solution that Fits your Project

by Jan Axelson

If you're designing something that will connect to a PC, chances are you'll consider using the Universal Serial Bus (USB). USB ports have been standard on new PCs for several years now, and the "legacy" serial and parallel ports that you could always count on in the past are beginning to disappear.

There are many ways to get a USB device up and running. In this article, I'll show three ways to add USB to a project. Each is best suited for a particular situation, so you can select the one that best fits your needs, skills, and budget.

This article first appeared in the September 2001 issue of Nuts & Volts magazine (www.nutsvolts.com).

I Just Want to Do a Little I/O

If your needs are basic, the USBI2CIO board from DeVaSys (Figure 1) is a ready-made solution that gives you 20 bits that you can use as inputs or outputs in any combination. The bits are standard 5V digital logic that can interface to switches, relay controls, LEDs, or other circuits. Windows applications can read and write to the bits. There's also an I^{2}C serial interface that connects to an on-board serial EEPROM.

Where to Buy the Books

Lakeview Research Info

Contact Lakeview

News for the Press

Magazine Articles

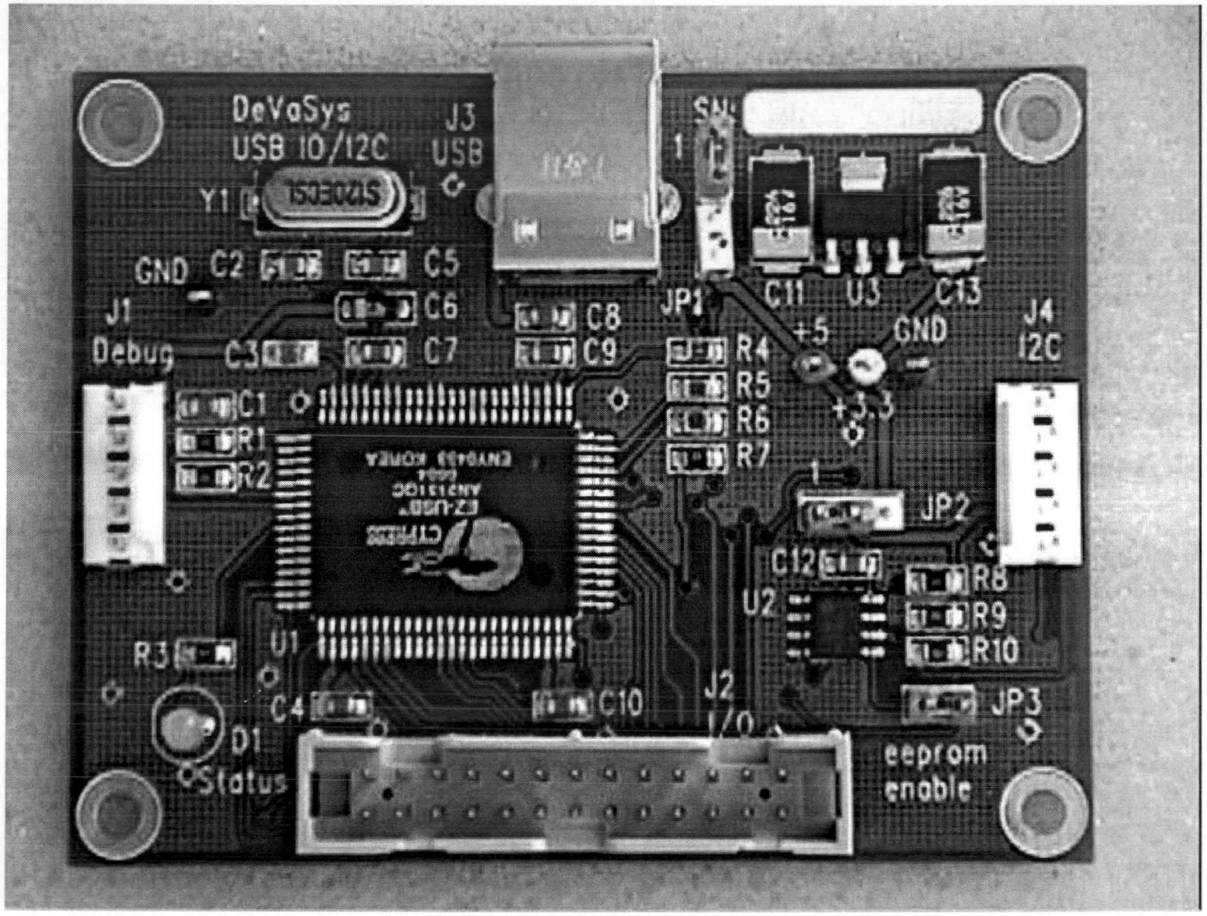

Figure 1. The USBI2CIO board from DeVaSys has everything you need to monitor and control 20 input/output bits via a USB port.

For the PC side of the link, DeVaSys provides several files, including a Windows device driver that is the link between the applications you write and the low-level drivers that control communications on the bus.

To read and write to the ports, the device driver supports the functions ReadIoPorts and WriteIoPorts. In my tests of the board in a compiled Visual Basic 6 application, each read or write of the 20 I/O bits took about 4 milliseconds, which works out to 250 reads or writes per second. This is plenty fast for many projects.

You can add other chips to the board's I^{2}C bus. For example, Philips Semiconductor's PCF8574 Remote 8-bit I/O expander has an I^{2}C interface and eight bits of parallel I/O. You can control this and other I^{2}C chips with the driver's ReadI2c and WriteI2c functions.

DeVaSys' website has example applications in Visual Basic and Visual C++ and a schematic of the board. The USB controller on the board is a Cypress AN2131 EZ-USB. The EZ-USB is an enhanced, 8051-compatible microcontroller with a full-speed USB interface. The board's 16-kilobyte I^{2}C EEPROM (a 24LC128) can store program code or data. The 3.3V regulator can receive power from the bus or an external 5V supply.

As shipped, the board contains no program code. Instead, the provided device driver uses the EZ-USB's ability to receive program code from the PC when the board attaches to the PC or the PC boots. This makes updates very easy because there are no EPROMs to program.

The board isn't limited to using the driver and device code that come with it. You can write your own program code and use other device drivers, including drivers provided with Windows. The EZ-USB chips are fast, flexible, and very capable. They even have two UARTs for RS-232 or similar links. Because of the support built into the chips, the amount of code you need to add for USB communications is less than what other chips require. A free assembler and a trial version of a C compiler are available from Cypress, along with complete documentation and sample code.

I Like PICs!

Microchip's PIC controllers are popular for good reason. They're inexpensive, easily available, and Microchip offers a variant for just about any purpose. Now there are two PICs that support USB, the PIC16C745 and PIC16C765 (Figure 2).

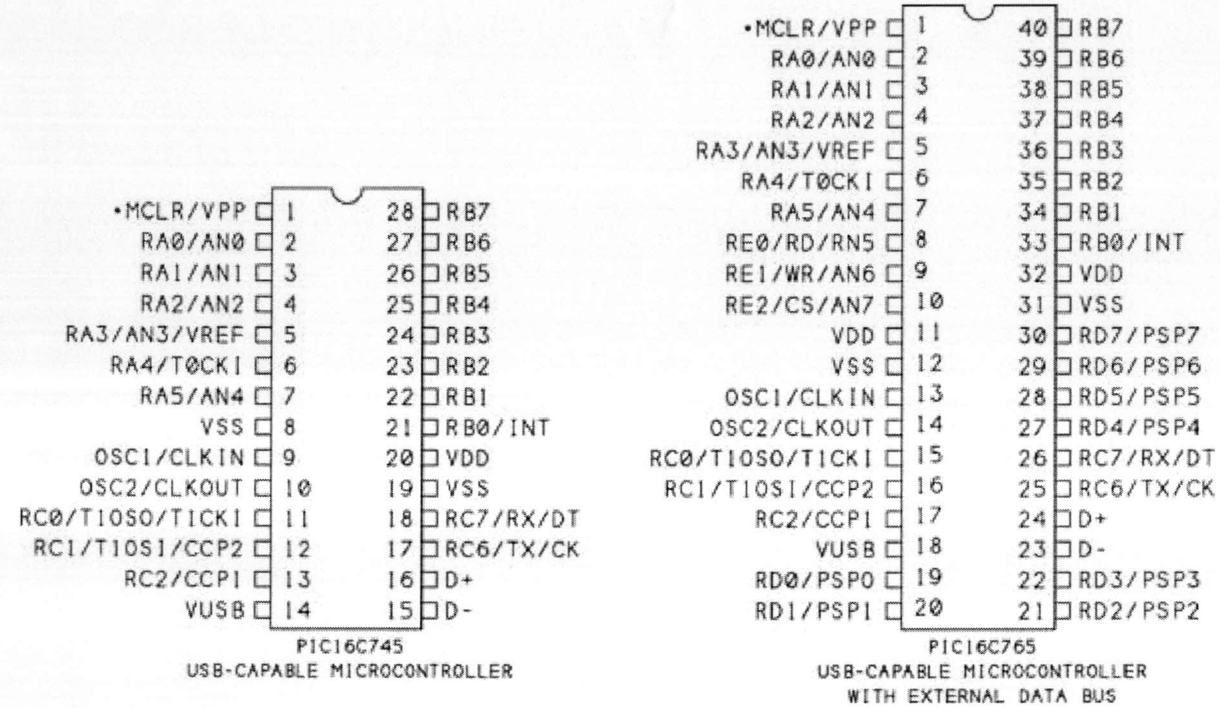

Figure 2. Microchip has two USB-capable PIC microcontrollers. Both have 19 I/O bits, and the '65 also has an external data bus. The USB data lines are D+ and D-. The VUSB pin provides power for the required pull-up resistor on D-.

Both chips support USB's low speed. In the world of USB, low speed means a bus rate of 1.5 Megabits per second, rather than full speed's 12 Megabits per second or USB 2.0's new high speed of 480 Megabits per second. To keep a single device from clogging the bus, the USB specification also limits the amount of bus time a low-speed device can reserve. At most, you can count on transferring 800 bytes per second in each direction.

The most popular use for low speed is devices in the human interface device (HID) class. HIDs include mice, keyboards, and joysticks, but you can also use the HID class for devices that don't fit into one of these categories. Windows includes drivers for HID communications, so if the code in your device meets the requirements of the HID spec, you don't have to provide a driver.

The PICs are enhanced members of Microchip's 16C5x series. Code written for the 16C5x is portable to the 16C7x5. Besides the USB interface, the chips have nineteen I/O pins, plus the '65 has an 8-bit parallel slave port for connecting to a microcontroller with an external data bus. Up to eight of the I/O pins can function as inputs to an on-chip analog-to-digital converter.

An on-chip USART supports two other kinds of serial links. With Maxim Semiconductor's MAX232 or a similar chip, you can use the USART for RS-232 communications. The USART also supports synchronous communications, where the interface includes a clock line. Either the PIC or an external source can provide the clock.

You can use a crystal or an inexpensive ceramic resonator to clock the CPU. Program memory is EPROM or one-time-programmable (OTP) PROM. The chips are available in through-hole and surface-mount packages. Microchip's MPLAB In-Circuit Emulator (ICE) 2000 development system supports the chips.

The PICs require a fair amount of code to support USB communications, including the code that responds to the standard requests the PC sends when the device attaches to the bus. Much of this code is the same for all applications, so adapting it for a specific project requires few changes. Microchip provides code for a HID-class device that sends and receives generic data.

Other microcontroller families also have USB-capable variants. If you have experience with a microcontroller, it makes sense to stick with it when you have a project with a USB interface. Fans of the 8051 can use Cypress' EZ-USB or Infineon's C541U. Atmel's AVR family includes the USB-capable AT76C711. For an 8-bit Motorola microcontroller, check out the 68HC05JB3/4 and 68HC08JB8.

I Want the Option to Use either RS-232 or USB

The RS-232 serial port is a good, general-purpose interface that has been with the PC since its beginning. There are thousands of different RS-232 peripherals in use. But RS-232 ports are beginning to disappear from PCs. Fortunately, just about any RS-232 device can be designed to use USB, and it's easier than you might think to make a device that supports both interfaces. This way, you can use the device with everything from an old DOS PC that doesn't support USB to the latest "legacy-free" Windows PC that lacks an RS-232 port (though DOS will require different software on the PC).

An easy way to add USB capability to an RS-232 device is to buy a converter that translates between the two interfaces. A device driver included with the converter causes Windows to treat the USB device the same as if it were using a straight RS-232 connection.

With the converter option, you can use the same software on the PC to communicate with the device, no matter which interface it's using. The only limitation to using a converter is that the PC's application software must use standard Windows functions for accessing COM ports (ReadFile, WriteFile) or a custom control such as Visual Basic's MSComm, which uses these functions internally. Inp and Out functions that write directly to a port address won't work. Converters are available from many sources, including B & B Electronics, which has a large selection of USB-related items.

For a more elegant way to do the same thing, the USB UART from FTDI makes it easy to build a converter into a project.

A typical UART converts between the serial data used by RS-232 and a CPU's parallel data. Besides data lines, a UART may also support RS-232's handshaking signals such as RTS and CTS. Just about all UARTs use 5V (or 3V) logic, so converting to RS-232's positive and negative voltages requires added components, such as a MAX232 chip.

The USB UART works in a similar way to other UARTs, except instead of converting to and from parallel data, it converts to and from USB (Figure 3). To adapt an RS-232 design for USB, you move the CPU's connections from the MAX232 or other RS-232 voltage converter to the matching pins on the USB UART. The device's internal code requires no changes because the device still thinks it's talking over an RS-232 link as usual.

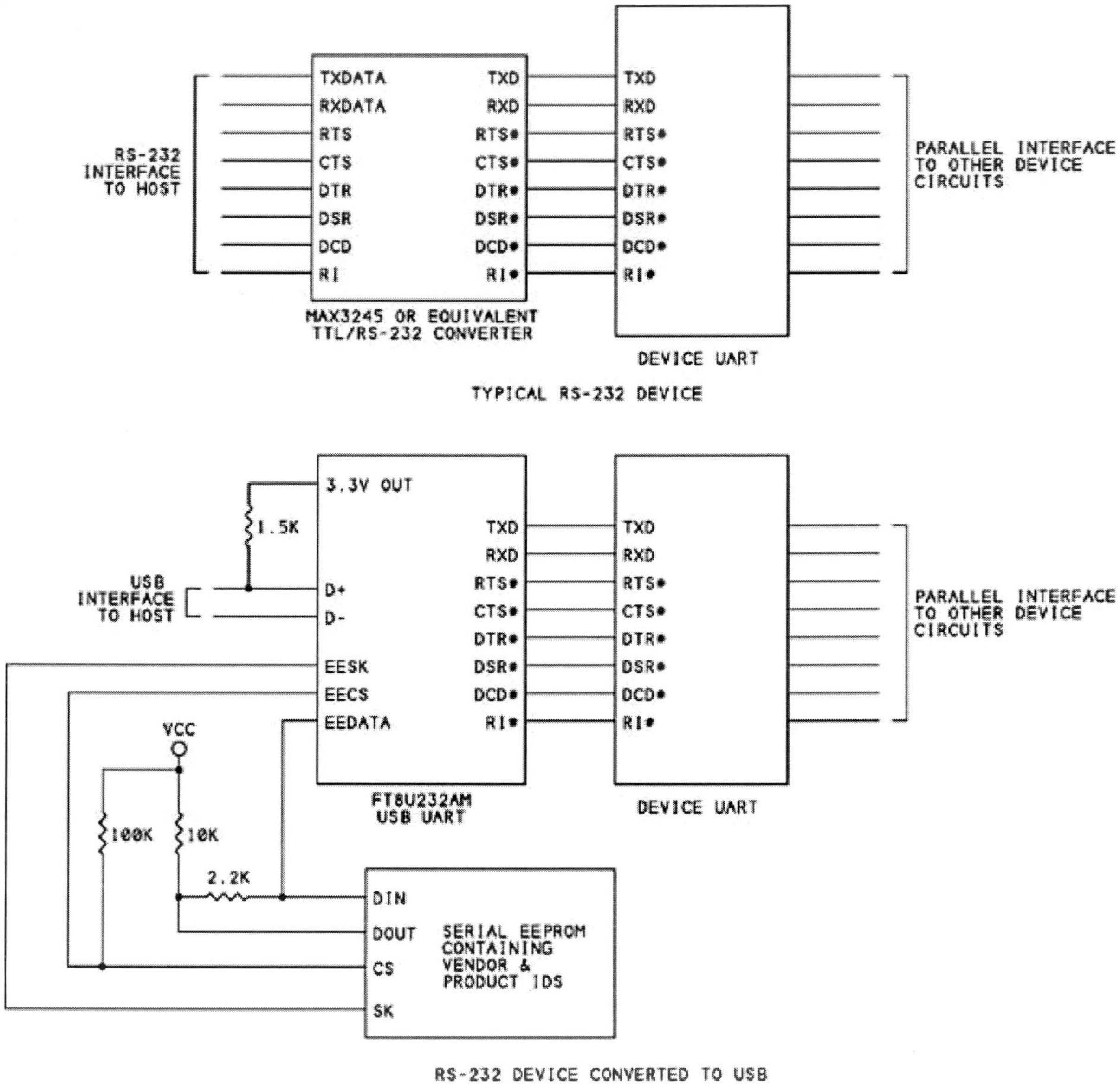

Figure 3. The USB UART from FTDI converts between RS-232 serial data and USB. The chip has pins to match each of RS-232's data and handshaking signals. A device connected to a USB UART thinks it's still communicating over RS-232.

The USB UART supports all of RS-232's handshaking signals, and FTDI provides a free device driver.

To support both interfaces, you can include a switch that routes the signals to the RS-232 converter or the USB UART. A circuit that detects a voltage on the USB's power line could switch the signals automatically, using the USB UART when the USB voltage is present and RS-232 otherwise.

One cost of USB developing is that any product you sell must contain a Vendor ID and Product ID. When a device attaches to a PC, Windows reads this information from the chip and uses it in locating the correct device driver. Vendor IDs are assigned by the USB Implementers Forum, which is the non-profit corporation that publishes the USB specification and in general supports USB development. A Vendor ID costs $1500 and includes the right to assign Product IDs associated with the Vendor ID.

The USB UART contains FTDI's Vendor ID and a Product ID. Because you don't change the programming in the chip, your product can use these values. You also have the option to store your own IDs in an external serial EEPROM. If you'd like a unique Product ID to use with FTDI's Vendor ID, you can request one free from FTDI.

The USB UART is a 32-lead surface-mount chip (a quad flatpack, or QFP). The chips are available in the U.S. from Saelig Company. For experimenting on a through-hole proto board, use a DIP (dual in-line package) adapter (Figure 4), though you'll need to do some fine soldering of the chip to the adapter. One source for adapters is www.smt-adapter.com.

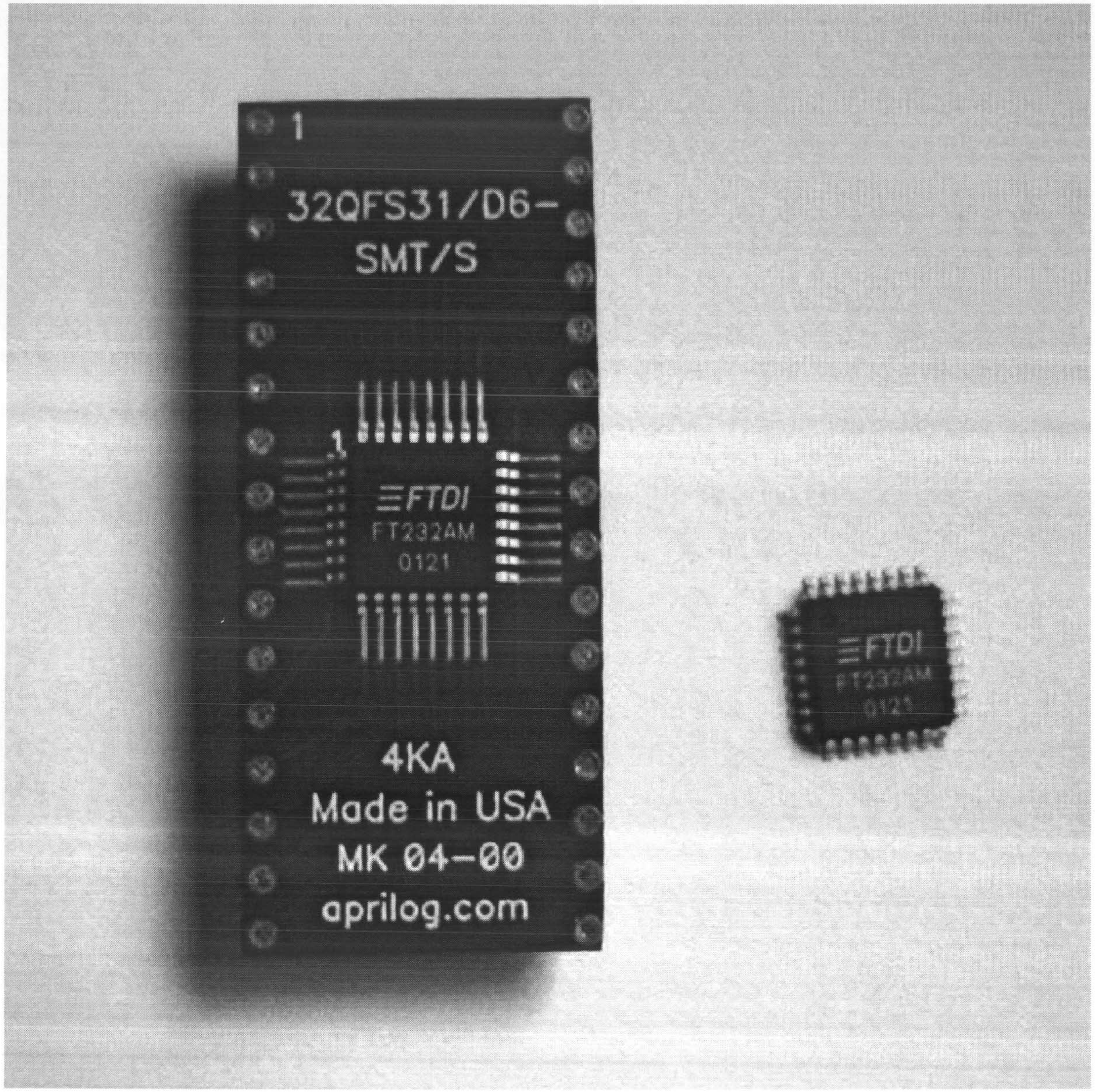

Figure 4. With an adapter, you can use the USB UART on a breadboard or a protoboard designed for through-hole components.

For another approach to USB using a different FTDI chip, see "Add a Universal Serial Bus Interface to your Next Project" by Don L. Powrie in the May 2001 issue of *Nuts & Volts*.

Jan Axelson is the author of the just-released Second Edition of *USB Complete: Everything You Need to Develop Custom USB Peripherals*. Visit Jan's USB Central web page at *www.Lvr.com/usb.htm*.

Sidebar: What about USB 2.0's New High Speed?

One of the biggest news items relating to USB is its new high speed of 480 Megabits per second. High speed became a possibility with the release of version 2.0 of the USB specification in 2000. Making high speed a reality requires three things: a USB 2.0 host controller in the PC, support for USB 2.0 under Windows, and a device that contains a high-speed-capable controller.

USB 2.0 host controllers will soon be standard on new motherboards, or you can add a controller to an existing system on a card that fits in a PCI slot. Microsoft has promised that support for USB 2.0 will either be included in the latest Windows edition, called Windows XP, or it will be available as an update soon afterwards. Updates for Windows 2000 and Windows Me are likely to follow. High-speed controllers for peripherals are available now from a few vendors, and the selection will increase in time.

What about all of the older, slower USB 1.x hardware? Just about all of the high-speed peripherals will also function at full speed, so they'll be usable with older hardware. And the 2.0 controllers are backwards-compatible, so all of your full- and low-speed peripherals will still work fine with the new controllers.

Web Links

Atmel USB-capable AVR controller (AT76C711)
www.atmel.com

B & B Electronics

USB/RS-232 converters
www.bb-elec.com

Cypress Semiconductor EZ-USB
www.cypress.com

DeVaSys USBI2CIO board
www.devasys.com

DIP adapter board for FTDI USB UART
Part #32qfs31/D6-SMT/S
www.smt-adapter.com

FTDI USB UART
www.ftdichip.com/

Infineon 8051-compatible, USB-capable microcontroller (C541U)
www.infineon.com

Microchip PIC16C745 and PIC16C765 USB microcontrollers
www.microchip.com

Motorola 68HC05JB3/4 and 68HC08JB8 USB microcontrollers
www.motorola.com

Philips Semiconductor's PCF8574 Remote 8-bit I/O expander
www.philips.com

Saelig Company
U.S. distributor of the USB UART
www.saelig.com

Jan Axelson's USB Central
Tutorials, example code, and other info for USB developers

www.lvr.com/usb.htm